UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/595,776	06/16/2000	Enric Musoll	P3810	1143

23669 7590 12/14/2005

HUFFMAN LAW GROUP, P.C.
1832 N. CASCADE AVE.
COLORADO SPRINGS, CO 80907-7449

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 12/14/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/595,776	MUSOLL ET AL.	
	Examiner	Art Unit	
	David J. Huisman	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *30 September 2005*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *22 February 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *5/19, 6/2, & 9/29*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-20 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: IDS as received on 4/23/2005, 5/19/2005, and 9/29/2005; Request For Corrected Filing Receipt as received on 6/1/2005, and Extension of Time and Amendment as received on 9/30/2005.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Claims 6-10 and 17 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

5. Claim 6 recites the limitation "said fetch algorithm" in the last paragraph. There is insufficient antecedent basis for this limitation in the claim. For purposes of examination, it will be interpreted as "said fetch stage."

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 1-15 and 18-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Yoaz et al., "Speculative Techniques for Improving Load Related Scheduling," May 1999 (as applied in the previous Office Action and herein referred to as Yoaz), in view of Hoyt et al., U.S. Patent No. 5,604,877 (herein referred to as Hoyt). In addition, Parady, U.S. Patent No. 5,933,627 (as applied in the previous Office Action and herein referred to as Parady) is cited as extrinsic evidence for showing that it is common to have separate hardware streams for each thread.

8. Referring to claim 1, Yoaz has taught in a processor having multiple hardware streams supporting multiple data threads, and a data cache, a system for fetching instructions from one to P of the multiple hardware streams to a pipeline, where P is less than the number of multiple hardware streams (for purposes of this examination, P is interpreted as being equal to 1), the system comprising:

a) multiple hit/miss predictors, each associated with a corresponding one of the multiple hardware streams, said each configured to forecast whether corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss the data cache. See page 47, column 1, and note the paragraph beginning with "Another...". Clearly, multiple threads exist, and a predictor is used for each thread. Consequently, there are multiple predictors. Also, the forecasts of these predictors would be used by the fetching hardware (if predicted to hit, continue fetching from same stream; otherwise, switch and fetch from another stream).

b) a fetch stage, coupled to said multiple hit/miss predictors, configured to simultaneously fetch every cycle, the instructions from the one to P of the multiple hardware streams to the pipeline, and configured to select, on a cycle-by-cycle basis, the one to P of the multiple hardware streams from which to fetch the instructions. Again, see page 47, column 1, and note the paragraph beginning with "Another...". The fetch algorithm includes the steps of fetching from a current thread as long as a miss is not predicted, and when a miss is predicted, switching to and fetching instructions from a second thread. It should be realized that this happens on a cycle-by-cycle basis, as instructions are executed every cycle. Whenever a load instruction appears, the prediction must be made, and if multiple loads occur in a row then the predictions are made for those consecutive cycles.

c) Yoaz has not explicitly taught that said multiple hit/miss predictors forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss the data cache prior to when said corresponding instructions enter into a dispatch stage in the pipeline. However, Hoyt has taught the concept of making predictions for instructions before the instructions are fetched (and also before any dispatch stage). More specifically, the address associated with the instruction (the PC) is applied to a predictor in order to make a prediction fast enough such that the results of the prediction are seen and used by the system as soon as possible. See column 8, lines 41-53. A person of ordinary skill in the art would have recognized that a similar pre-dispatch prediction scheme would be useful in Yoaz because such a scheme would allow a load miss to be predicted as soon as possible, and consequently, a new stream would be selected as soon as possible, thereby reducing any flushing and/or delay associated with a delayed prediction. The implementation of Hoyt's prediction system in Yoaz would result

in predicting a load miss when it is time to fetch the load. If a load is predicted to miss while it is being fetched, then in the very next cycle a new stream may be fetched, thereby eliminating any delay. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to forecast cache-miss predictions for instructions prior to those instructions entering a dispatch stage.

9. Referring to claim 2, Yoaz in view of Hoyt has taught a system as described in claim 1. Yoaz has further taught that a hit prediction precipitates no change in the fetching of the instructions. See page 47, column 1, and note the paragraph beginning with "Another...". It should be realized that Yoaz only discloses thread switching based on a predicted cache miss because that is when an expensive main memory access is likely required. When there is a hit in the cache, no main memory access is required and therefore, no switch needs to be made.

10. Referring to claim 3, Yoaz in view of Hoyt has taught a system as described in claim 1. Yoaz has further taught that a miss prediction results in switching the fetching to different ones of the multiple hardware streams. See page 47, column 1, and note the paragraph beginning with "Another...". When a miss prediction is made, a thread switch occurs. That is, the system will begin fetching a new thread instead of sitting idle while the current thread finishes the time-consuming main memory access. A first switch may result in switching to a stream B, while another switch may result in switching to a stream C (different ones). Each stream is a flow of data from a source to a sink. That is from a first thread to the execution units is a first stream, a second thread to the execution units is a second stream, and so on. A hardware stream is nothing more than the hardware which supports this flow of instructions. In addition, Parady has shown that multiple hardware streams exist for multiple threads. See Fig.3, and note that the wires

coupling the dispatch unit and each of the instruction buffers forms a multiple hardware streams, one for each thread, in which thread instructions may flow.

11. Referring to claim 4, Yoaz in view of Hoyt has taught a system as described in claim 1. Yoaz has further taught that said each of said multiple hit/miss predictors generates a confidence level value, and said confidence level value is used by said fetch algorithm to select the P of the one of the multiple hardware streams. See page 47, column 1, and note the paragraph beginning with "Another...". Clearly, the predictors will predict a miss to occur or not to occur. This prediction is a confidence level value in that if a miss is predicted to occur, then the predictor is confident that the current thread will miss the cache, whereas if a miss is not predicted to occur, then the predictor is confident that the current thread will hit the cache. These confidence values are used to determine whether fetching will continue from the current thread or whether a thread switch occurs and fetching will begin from a new thread.

12. Referring to claim 5, Yoaz in view of Hoyt has taught a system as described in claim 1. Yoaz has further taught that a said multiple hit/miss predictors further operate at a dispatch level to optimize the dispatch of consumer instructions by predicting latency of data. See page 47, column 1, and note the paragraph beginning with "Another...". It should be noted that the predictions determine which thread will be fetched from. Consequently, the predictors also determine which threads will be dispatched, as instructions which are fetched are also dispatched, otherwise they cannot be executed. And, it is known that consumer instructions are instructions which merely access some register or memory location to perform the desired operation, and therefore, it is expected that consumer instructions will be dispatched. In addition, the predictors predict the latency of data in that if a miss is predicted, then it is

predicted that it will take a long latency to retrieve the data (since main memory is accessed). However, if a miss is not predicted, then it is predicted that a short latency is required as the data will be available in the cache.

13. Referring to claim 6, Yoaz has taught a processor having multiple hardware streams supporting multiple data threads, the processor comprising:

a) a data cache, comprising a plurality of levels. See page 47, column 1, Fig.3, and the paragraphs beginning with "Employing..." and "Another...". It is clear that there is an L1 and L2 cache.

b) multiple hit/miss predictors, each associated with a corresponding one of the multiple hardware streams, said each configured to forecast whether corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss said data cache (see page 47, column 1, and note the paragraph beginning with "Another...". Clearly, multiple threads exist, and a predictor is used for each thread. Consequently, there are multiple predictors. Also, each stream is a flow of data from a source to a sink. That is, from a first thread to the execution units is a first stream, a second thread to the execution units is a second stream, and so on. A hardware stream is nothing more than the hardware which supports this flow of instructions (again, see Parady as described in the rejection of claim 1), said each of said multiple hit/miss predictors comprising:

b1) a plurality of hit/miss predictors, each configured to forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss one or more of said levels. See page 47, column 1, and note the

paragraph beginning with "Another...". Note that it is predicted if the thread will miss the L2 cache.

c) a fetch stage, coupled to said multiple hit/miss predictors, for simultaneously fetching every cycle, instructions from one to P of the multiple hardware streams, wherein P is less than the number of multiple hardware streams, and configured to select, on a cycle-by-cycle basis, said one to P of the multiple hardware streams from which to fetch said instructions, wherein said fetch stage selects said one to P of the multiple hardware streams based upon whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss said one or more of said levels. Again, see page 47, column 1, and note the paragraph beginning with "Another...". The fetch algorithm includes the steps of fetching from a current thread as long as a miss is not predicted, and when a miss is predicted, switching to and fetching instructions from a second thread. It should be realized that this happens on a cycle-by-cycle basis, as instructions are executed every cycle. Whenever a load instruction appears, the prediction must be made, and if multiple loads occur in a row then the predictions are made for those consecutive cycles.

d) Yoaz has not explicitly taught that said multiple hit/miss predictors forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss the data cache prior to when said corresponding instructions enter into a dispatch stage in a pipeline of the processor. However, Hoyt has taught the concept of making predictions for instructions before the instructions are fetched (and also before any dispatch stage). More specifically, the address associated with the instruction (the PC) is applied to a predictor in order to make a prediction fast enough such that the results of the prediction are seen and used by the

system as soon as possible. See column 8, lines 41-53. A person of ordinary skill in the art would have recognized that a similar pre-dispatch prediction scheme would be useful in Yoaz because such a scheme would allow a load miss to be predicted as soon as possible, and consequently, a new stream would be selected as soon as possible, thereby reducing any flushing and/or delay associated with a delayed prediction. The implementation of Hoyt's prediction system in Yoaz would result in predicting a load miss when it is time to fetch the load. If a load is predicted to miss while it is being fetched, then in the very next cycle a new stream may be fetched, thereby eliminating any delay. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to forecast cache-miss predictions for instructions prior to those instructions entering a dispatch stage.

14. Referring to claim 7, Yoaz in view of Hoyt has taught a processor as described in claim 6. Yoaz has further taught that a hit prediction precipitates no change in the fetching of said instructions. See page 47, column 1, and note the paragraph beginning with "Another...". It should be realized that Yoaz only discloses thread switching based on a predicted cache miss because that is when an expensive main memory access is likely required. When there is a hit in the cache, no main memory access is required and therefore, no switch needs to be made.

15. Referring to claim 8, Yoaz in view of Hoyt has taught a processor as described in claim 8. Yoaz has further taught that a miss prediction results in switching the fetching to different ones of the multiple hardware streams. See page 47, column 1, and note the paragraph beginning with "Another...". When a miss prediction is made, a thread switch occurs. A first switch may result in switching to a stream B, while another switch may result in switching to a stream C (different ones). That is, the system will begin fetching a new thread instead of sitting idle while

the current thread finishes the time-consuming main memory access. Each stream is a flow of data from a source to a sink. That is from a first thread to the execution units is a first stream, a second thread to the execution units is a second stream, and so on. A hardware stream is nothing more than the hardware which supports this flow of instructions.

16. Referring to claim 9, Yoaz in view of Hoyt has taught a processor as described in claim 6. Yoaz has further taught that said each of said multiple hit/miss predictors generates a confidence level value, and said confidence level value is used by said fetch stage to select said one to P of the multiple hardware streams. See page 47, column 1, and note the paragraph beginning with "Another...". Clearly, the predictors will predict a miss to occur or not to occur. This prediction is a confidence level value in that if a miss is predicted to occur, then the predictor is confident that the current thread will miss the cache, whereas if a miss is not predicted to occur, then the predictor is confident that the current thread will hit the cache. These confidence values are used to determine whether fetching will continue from the current thread or whether a thread switch occurs and fetching will begin from a new thread.

17. Referring to claim 10, Yoaz in view of Hoyt has taught a system as described in claim 6. Yoaz has further taught that a said multiple hit/miss predictors further operate at a dispatch level to optimize the dispatch of consumer instructions by predicting latency of data. See page 47, column 1, and note the paragraph beginning with "Another...". It should be noted that the predictions determine which thread will be fetched from. Consequently, the predictors also determine which threads will be dispatched, as instructions which are fetched are also dispatched, otherwise they cannot be executed. And, it is known that consumer instructions are instructions which merely access some register or memory location to perform the desired

operation, and therefore, it is expected that consumer instructions will be dispatched. In addition, the predictors predict the latency of data in that if a miss is predicted, then it is predicted that it will take a long latency to retrieve the data (since main memory is accessed). However, if a miss is not predicted, then it is predicted that a short latency is required as the data will be available in the cache.

18. Referring to claim 11, Yoaz has taught in a processor having multiple hardware streams supporting multiple data threads, and a data cache, a method for simultaneously fetching instructions every cycle from one to P of the multiple hardware streams to a pipeline, where P is less than the number of the multiple hardware streams (note that P=1, and P<X, where X is all of the available streams from which instructions may be fetched), the method comprising:

a) for each of the multiple hardware streams, making a hit/miss prediction by a corresponding one of associated hit/miss predictors as to whether corresponding instructions for the each of the multiple hardware streams previously fetched will hit or miss the cache. See page 47, column 1, and note the paragraph beginning with "Another...". Clearly, multiple threads exist, and a predictor is used for each thread. Consequently, there are multiple predictors.

b) selecting, on a cycle-by-cycle basis, the one to P of the multiple hardware streams from which to fetch the instructions. Again, see page 47, column 1, and note the paragraph beginning with "Another...". The method includes the steps of fetching from a current thread (P=1) as long as a miss is not predicted, and when a miss is predicted, switching to and fetching instructions from a second thread. It should be realized that this happens on a cycle-by-cycle basis, as instructions are executed every cycle. Whenever a load instruction appears, the prediction must be made, and if multiple loads occur in a row then the predictions are made for those consecutive cycles.

c) Yoaz has not explicitly taught that said making is performed prior to when the corresponding instructions enter into a dispatch stage in the pipeline. However, Hoyt has taught the concept of making predictions for instructions before the instructions are fetched (and also before any dispatch stage). More specifically, the address associated with the instruction (the PC) is applied to a predictor in order to make a prediction fast enough such that the results of the prediction are seen and used by the system as soon as possible. See column 8, lines 41-53. A person of ordinary skill in the art would have recognized that a similar pre-dispatch prediction scheme would be useful in Yoaz because such a scheme would allow a load miss to be predicted as soon as possible, and consequently, a new stream would be selected as soon as possible, thereby reducing any flushing and/or delay associated with a delayed prediction. The implementation of Hoyt's prediction system in Yoaz would result in predicting a load miss when it is time to fetch the load. If a load is predicted to miss while it is being fetched, then in the very next cycle a new stream may be fetched, thereby eliminating any delay. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to forecast cache-miss predictions for instructions prior to those instructions entering a dispatch stage.

19. Referring to claim 12, Yoaz in view of Hoyt has taught a method as described in claim 11. Yoaz has further taught that said making comprises generating a confidence level value, and employing the confidence level value to select the one to P of the multiple hardware streams. See page 47, column 1, and note the paragraph beginning with "Another...". Clearly, the predictors will predict a miss to occur or not to occur. This prediction is a confidence level value in that if a miss is predicted to occur, then the predictor is confident that the current thread will miss the cache, whereas if a miss is not predicted to occur, then the predictor is confident that the

current thread will hit the cache. These confidence values are used to determine whether fetching will continue from the current thread or whether a thread switch occurs and fetching will begin from a new thread.

20. Referring to claim 13, Yoaz in view of Hoyt has taught a method as described in claim 11. Yoaz has further taught further operating the multiple hit/miss predictors at a dispatch level to optimize the dispatch of consumer instructions by predicting latency of data. See page 47, column 1, and note the paragraph beginning with "Another...". It should be noted that the predictions determine which thread will be fetched from. Consequently, the predictors also determine which threads will be dispatched, as instructions which are fetched are also dispatched, otherwise they cannot be executed. And, it is known that consumer instructions are instructions which merely access some register or memory location to perform the desired operation, and therefore, it is expected that consumer instructions will be dispatched. In addition, the predictors predict the latency of data in that if a miss is predicted, then it is predicted that it will take a long latency to retrieve the data (since main memory is accessed). However, if a miss is not predicted, then it is predicted that a short latency is required as the data will be available in the cache.

21. Referring to claim 14, Yoaz in view of Hoyt has taught a method as described in claim 11. Yoaz has not explicitly taught that the processor comprises a fine-grained multistreaming processor that concurrently executes the instructions from the multiple hardware streams. However, Official Notice is taken that fine-grained multithreading (FMT) is well known and expected in the art. FMT allows expedites the completion of all of the threads being worked on, as every cycle, a next thread is switched in. Consequently, overall throughput is increased. And,

it should be realized that Yoaz has taught that the prediction method will work for multithreaded systems, without explicitly saying what types (FMT, CMT, etc.). A person of ordinary skill in the art would have recognized that even with FMT, load instructions will be executed, and therefore, this prediction scheme will be useful in avoiding switching back to the thread that is predicted to miss the cache. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to be a fine-grained multistreaming system so that the advantages of an FMT system may be achieved.

22. Referring to claim 15, Yoaz in view of Hoyt has taught a system as described in claim 1. Yoaz has further taught that the data cache comprises a first level and a second level (see page 47, column 1, Fig.3, and the paragraphs beginning with "Employing..." and "Another...". It is clear that there is an L1 and L2 cache.), and wherein said each of said multiple hit/miss predictors comprises:

a) a hit/miss predictor, configured to forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss said first level. See page 47, column 1, and note the paragraph beginning with "Another...". Note that if a miss is predicted at the second level of cache (L2), then a miss is inherently predicted at level one (L1). This is inherent because the nature of cache accessing forces it to be. More specifically, if there is no miss at level one, there is no need to access level two. Therefore, in order for a miss to matter at level two, there must have first been a miss at level one.

b) a hit/miss predictor, configured to forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss said second level. See page 47, column 1, and note the paragraph beginning with "Another...".

c) Yoaz has not explicitly taught separate first and second predictors for predicting first and second level cache misses, respectively. However, as shown in Nerwin v. Erlichman 168 USPQ 177 (1969), to make separable is generally not given patentable weight or would have been an obvious improvement. Doing would allow a user to customize each predictor, separately. Consequently, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz's hit/miss predictor into first and second predictors for predicting first and second level cache misses, respectively.

d) wherein said fetch stage selects the one to the P of the multiple hardware streams based upon whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss said second level. Again, see page 47, column 1, and note the paragraph beginning with "Another...". The fetch algorithm includes the steps of fetching from a current thread as long as a miss is not predicted, and when a miss is predicted, switching to and fetching instructions from a second thread.

23. Referring to claim 18, Yoaz in view of Hoyt has taught a method as described in claim 11. Yoaz has further taught that said selecting comprises switching the fetching to a different one to P of the multiple hardware streams. See page 47, column 1, and note the paragraph beginning with "Another...".

24. Referring to claim 19, Yoaz in view of Hoyt has taught a method as described in claim 11. Yoaz has further taught that the data cache comprises a first level and a second level (see page 47, column 1, Fig.3, and the paragraphs beginning with "Employing..." and "Another...". It is clear that there is an L1 and L2 cache.), and wherein said making comprises:

a) first forecasting whether said corresponding instructions from the corresponding one of the multiple hardware streams will hit or miss the first level. See page 47, column 1, and note the paragraph beginning with "Another...". Note that if a miss is predicted at the second level of cache (L2), then a miss is inherently predicted at level one (L1). This is inherent because the nature of cache accessing forces it to be. More specifically, if there is no miss at level one, there is no need to access level two. Therefore, in order for a miss to matter at level two, there must have first been a miss at level one.

b) second forecasting whether the corresponding instructions from the corresponding one of the multiple hardware streams will hit or miss the second level. See page 47, column 1, and note the paragraph beginning with "Another...".

c) wherein said selecting comprises choosing the one to P of the multiple hardware streams based upon whether the corresponding instructions from the corresponding one of the multiple hardware streams will hit or miss the second level. Again, see page 47, column 1, and note the paragraph beginning with "Another...". The fetch algorithm includes the steps of fetching from a current thread as long as a miss is not predicted, and when a miss is predicted, switching to and fetching instructions from a second thread.

25. Claims 16-17 and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Yoaz in view of Hoyt, as applied above, and further in view of Ryan, U.S. Patent No. 5,694,572 (as applied in the previous Office Action).

26. Referring to claim 16, Yoaz in view of Hoyt has taught a system as described in claim 1. Yoaz has further taught a network processor (Yoaz's processor may be used in any way

including in a network), but has not taught that said each of said multiple hit/miss predictors employs a flow number to which a packet belongs to forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss the data cache. However, Ryan has taught the concept of operating on first process packets (instructions and data) wherein the cache would be populated with data specifically for the first process. Consequently, when a switch is made to a next process (and group of packets), the cache will not be optimized because it is populated with data from the first process. Therefore, the next process will result in many more cache misses until the cache is populated. See column 2, lines 1-16. As a result, one way a prediction could be made is based on process numbers or flow numbers (note that process numbers are well known and expected in the art as they are used to distinguish processes) because a new flow number would mean that the cache is not optimized for that new flow number and a miss is much more likely. This would be a simple way of making a prediction. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to use flow numbers assigned to specific packets for making predictions.

27. Referring to claim 17, Yoaz in view of Hoyt has taught a processor as described in claim 6. Yoaz has further taught a network processor (Yoaz's processor may be used in any way including in a network), but has not taught that said each of said multiple hit/miss predictors employs a flow number to which a packet belongs to forecast whether said corresponding instructions from said corresponding one of the multiple hardware streams will hit or miss the data cache. However, Ryan has taught the concept of operating on first process packets (instructions and data) wherein the cache would be populated with data specifically for the first

process. Consequently, when a switch is made to a next process (and group of packets), the cache will not be optimized because it is populated with data from the first process. Therefore, the next process will result in many more cache misses until the cache is populated. See column 2, lines 1-16. As a result, one way a prediction could be made is based on process numbers or flow numbers (note that process numbers are well known and expected in the art as they are used to distinguish processes) because a new flow number would mean that the cache is not optimized for that new flow number and a miss is much more likely. This would be a simple way of making a prediction. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to use flow numbers assigned to specific packets for making predictions.

28. Referring to claim 20, Yoaz in view of Hoyt has taught a method as described in claim 11. Yoaz in view of Hoyt has not taught that said making comprises employing a flow number to which a packet belongs to forecast whether the corresponding instructions from the corresponding one of the multiple hardware streams will hit or miss the data cache.. However, Ryan has taught the concept of operating on first process packets (instructions and data) wherein the cache would be populated with data specifically for the first process. Consequently, when a switch is made to a next process (and group of packets), the cache will not be optimized because it is populated with data from the first process. Therefore, the next process will result in many more cache misses until the cache is populated. See column 2, lines 1-16. As a result, one way a prediction could be made is based on process numbers or flow numbers (note that process numbers are well known and expected in the art as they are used to distinguish processes) because a new flow number would mean that the cache is not optimized for that new flow

number and a miss is much more likely. This would be a simple way of making a prediction. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Yoaz to use flow numbers assigned to specific packets for making predictions.

Response to Arguments

29. Applicant's arguments with respect to claims 1-20 have been considered but are moot in view of the new ground(s) of rejection. However, it should be pointed out that the examiner has been unable to locate any part of Yoaz which teaches away from having prediction prior to a dispatch stage. There is no explicit pipeline structure set forth. Hoyt is introduced to show that predictions may be made in the fetch stage (before the dispatch stage). Even though Hoyt is concerned with branches, a similar system may be employed in Yoaz. Whether an instruction is a load or branch, a prediction may be made early. The outcome of both types of predictions is to choose a particular stream of instructions to fetch from. In the case of Yoaz, a load prediction will either result in fetching from the same stream (hit prediction) or a new stream (miss prediction). Likewise, in Hoyt, a branch prediction will either result in fetching from the same stream (not taken prediction) or a new stream (taken prediction). Earlier predictions result in earlier detection of which stream to fetch from, which would thereby reduce delay or any necessary flushing associated with predictions. So, in general, the examiner asserts that the Hoyt teachings are analogous to Yoaz.

Conclusion

30. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (571) 272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

DJH
David J. Huisman
November 23, 2005

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100